TD1 Asm Cortex-M3
Premier sujet de TD

3 IMACS 2010-2011

Vincent MAHOUT

***Résumé*—Le but de ce TD est tout d'abord de comprendre un petit bout de code. Cela suppose donc de voir comment l'enchaînement d'instructions basiques permettent de construire un algorithme simple. Une aspect important de compréhension concerne l'accès en mémoire par indirection. Enfin, il s'agira de jouer sur ces modes d'adressage afin de modifier le code dans l'espoir de l'optimiser.**

I. Le programme

Soit le morceau de programme suivant :

```
59
60          LDR R0,= 0x20000006
61          MOV R1,#0
62          MOV R3,#0
63  Calcul  LDRB R2,[R0]
64          ADD R1,R2
65          ADD R0,#1
66          ADD R3,#1
67          CMP R3,#10
68          BEQ Calcul
69
70  inf     B inf       ; boucle infinie
```

FIGURE 1. Premier listing

Lors de l'exécution de ce code la mémoire aura le contenu suivant :

II. comprehension

Dans un premier temps, il s'agit d'analyser ce programme.

Commentaire : leur faire reprendre la notion du test conditionné....vu rapidement en cours....insister sur le fait qu'un cmp tout seul ne sert à rien

Question 1: Que font les deux lignes *CMP R3,#10* et *BEQ Calcul* ? - Que peut -on en déduire sur le déroulement du programme ?

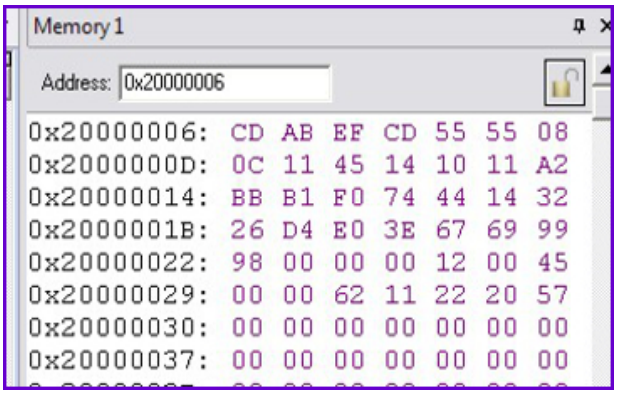

FIGURE 2. Contenu mémoire

Commentaire : la structuration algo n'est pas encore vue en cours - faire extraire un organigramme pour introduire cela....

Question 2: Déterminez ce que fait ce petit programme en extrayant son organigramme général.

III. Adressage mémoire

Commentaire : adressage indirect assez détaillé en cours. Cela ne doit poser pb qu'aux absents ! ! !

Question 3: Remplissez le tableau relatant l'état des registres *R0* à *R3* (en hexadécimal) à chaque fois que *PC* passe sur l'étiquette *Calcul*.

	R0	R1	R2	R3
1				
2				
3				
4				
5				
6				
7				
8				
9				
10				

Commentaire : reprendre la notion qu'en assembleur il n'y a pas de type de données jsute des quantités d'octets à charger => modiifcaiton de l'incrémentation du pointeur ...Attention un int est a priori signé donc l'instruction de chargement est LDRSH ...

Question 4: Le programmeur s'est trompé dans son codage. En effet, le tableau contient 5 *short int* (2 octets) et non 10 *char* (1 seul octet). Modifiez les instructions pour apporter cette correction.

Commentaire : reprendre l'inversion pf PF de sdonnes en mémoire pour faire l'addition correctement

Question 5: Cette modification prise en compte, que vaudra le registre *R1* lorsque *PC* pointera sur l'étiquette *inf* ?

IV. Optimisation du code

Commentaire : reprendre le post déplacement...et son intérêt pour ne plus gérer l'incrémentation du pointeur - 1 ligne de code en mois

Question 6: Modifiez la gestion du pointeur *R0* pour utiliser un adressage indirect avec post-déplacement.

Commentaire : il n'y a pas de codage unique - décrémenter plutôt qu'incrémenter est souvent plu sjudicieix car cela permet d'utiliser le drapeau Z

Question 7: Modifiez le code pour supprimer l'instruction *CMP* en se rappelant que l'instruction *SUBS* modifie l'état du fanion *Z*.